World Academy of Science, Engineering and Technology
International Journal of Electronics and Communication Engineering
Vol:1, No:11, 2007

Performance Evaluation of Low Density Parity Check Codes

Othman O. Khalifa, Sheroz khan, Mohamad Zaid, and Muhamad Nawawi

Abstract—This paper mainly about the study on one of the widely used error correcting codes that is Low parity check Codes (LDPC). In this paper, the Regular LDPC code has been discussed The LDPC codes explained in this paper is about the Regular Binary LDPC codes or the Gallager.

***Keywords*—**LDPC, channel coding.

I. Introduction

CODING for error correction is one of the many tools available for achieving reliable data transmission in communication system[4].Low-density parity-check (LDPC) codes are a class of linear block LDPC codes[4]. LPDC codes were re-discovered independently by Mackay and Neal and Wiberg[1].The name comes from the characteristic of their parity-check matrix which contains only a few 1's in comparison to the amount of 0's. Their main advantage is that they provide a performance which is very close to the capacity for a lot of different channels and linear time complex algorithms for decoding. Furthermore are they suited for implementations that make heavy use of parallelism.

A. History

Low-density parity-check (LDPC) codes are forward error-correction codes, first proposed in the 1962 PhD thesis of Gallager at MIT, but then largely neglected for over 35 years. In the mean time the field of forward error correction was dominated by highly structured algebraic block and convolutional codes. Despite the enormous practical success of these codes, their performance fell well short of the theoretically achievable limits set down by Shannon in his seminal 1948 paper [1]. By the late 1980s, researchers were largely resigned to this seemingly insurmountable theory–practice gap. The situation was utterly revolutionized by "turbo codes," proposed by Berrou, Glavieux and Thitimajshima in 1993,

In 2006, it is generally recognized that message-passing decoders operating on sparse graphs are the "right" way of thinking about high-performance error correction. Moreover, the "soft/iterative" paradigm introduced by Gallager, that some researchers refer to as the "turbo principle", has been extended to a whole raft of telecommunications problems, including channel equalization, channel estimation, and source coding.

Authors are with Department of Electrical and Computer Engineering, International Islamic University Malaysia (e-mail: khalifa@iiu.edu.my).

II. Literature Review

A. LDPC Codes

Generally there are two different methods to represent LDPC codes. Like all linear block codes they can be described via matrices. The second method is a graphical representation.

1. Matrix Representation

Let's look at an example for a low-density parity-check matrix first. The matrix defined in equation (1) is a parity check matrix with dimension n ×m for a (8, 4) code. We can now define two numbers describing these matrixes. wr for the number of 1's in each row and wc for the columns. For a matrix to be called low-density the two conditions wc << n and wr << m must be satisfied. In order to do this, the parity check matrix should usually be very large, so the example matrix can't be really called low-density.

$$\mathbf{H} = \begin{bmatrix} 0 & 1 & 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \end{bmatrix} \quad (1)$$

2. Graphical Representation

Tanner introduced an effective graphical representation for LDPC Tanner codes. Tanner graphes are bipartite graphes. That means that the nodes of the graph are separated into two distinctive sets and edges are only connecting nodes of two different types. The two types of nodes in a Tanner graph are called variable nodes (v-nodes) and check nodes (c-nodes). Fig. 1.is an example for such a Tanner graph and represents the same code as the matrix in 1. The creation of such a graph is rather straight forward. It consists of m check nodes (the number of parity bits) and n variable nodes (the number of bits in a codeword). Check node fi is connected to variable node cj if the element hij of H is a 1.

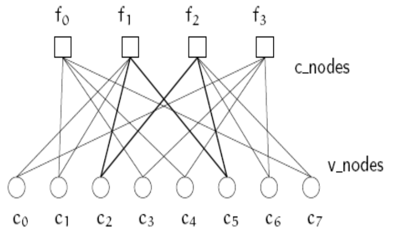


Fig. 1 Tanner graph corresponding to the parity check matrix in equation (1). The marked path c2 ! f1 ! c5 ! f2 ! c2 is an example for a short cycle. Those should usually be avoided since they are bad for decoding performance

B. Regular and Irregular LDPC Codes

A LDPC code is called regular if wc is constant for every column regular and wr = wc · (n/m) is also constant for every row. The example matrix from equation (1) is regular with wc = 2 and wr = 4. Gallager's construction technique:

a) Code parameters N, K, wc, wr are given.

b) Construct the following matrix with $\frac{N-K}{wc}$ rows and N columns:

c) Let π(H1) be a pseudo-random column permutation of H1.

d) Construct regular LDPC check matrix by stacking wc submatrices:

$$H=\left[\begin{array}{c} H\ 1 \\ \hline \Pi\ (H\ 1) \\ \hline \Pi\ (H\ 1) \\ \\ \end{array}\right]$$

Regular (Gallager) LDPC Code Example

This code has N=20, K = 5, wc =3, wr= 4.

$$H=\begin{array}{cccccccccccccccccccc}
\hline
1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&1&1&1\\
\hline
1&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&0\\
0&1&0&0&0&1&0&0&0&1&0&0&0&0&0&0&1&0&0&0\\
0&0&1&0&0&0&1&0&0&0&0&0&0&1&0&0&0&1&0&0\\
0&0&0&1&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0\\
0&0&0&0&0&0&0&1&0&0&0&1&0&0&0&1&0&0&0&1\\
\hline
1&0&0&0&0&1&0&0&0&0&0&1&0&0&0&0&0&1&0&0\\
0&1&0&0&0&0&1&0&0&0&1&0&0&0&0&1&0&0&0&0\\
0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&0&1&0\\
0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&1&0&0&0\\
0&0&0&0&1&0&0&0&0&1&0&0&0&0&1&0&0&0&0&1\\
\hline
\end{array}$$

The number of ones is *N.wc* = (*N–K*).wr. From this, it is easy to show that the rate of the code is:

$$R = 1 - \frac{wc}{wr}$$

In this example, R = 1 - 3/4 = 1/4.

There is the same number of incoming edges for every v-node and also for all the c-nodes. If H is low density but the numbers of 1's in each row or column aren't constant the code is called an irregular LDPC code.

C. Performance and Complexity

Before describing encoding and decoding algorithms in next section, we would like to explain why all this effort is needed. The feature of LDPC codes to perform near the Shannon limit1 of a channel exists only for large block lengths. For example there have been simulations that perform within 0.04 dB of the Shannon limit at a bit error rate of 10^{-6} with a block length of 10^{7}.The large block length results also in large parity-check and generator matrices. The complexity of multiplying a codeword with a matrix depends on the amount of 1's in the matrix. If we put the sparse matrix H in the form [H^{T} I] via Gaussian elimination the generator matrix G can be calculated as G = [I P]. The sub-matrix P is generally not sparse so that the encoding complexity will be quite high.

Since the complexity grows in O(n^{2}) even sparse matrices don't result in a good performance if the block length gets very high. So iterative decoding (and encoding) algorithms are used. Those algorithms perform local calculations and pass those local.

D. Encoding

A generator matrix for a code with parity-check matrix H can be found by performing Gauss-Jordan elimination on H to obtain it in the form

H = [A, In−k],

where A is a (n − k) × k binary matrix and In−k is the size n − k identity matrix. The generator matrix is then $G= [I_k, A^T]$.

Here we will go into this process in more detail using an example.

Example 1

We wish to encode the length 10 rate-1/2 LDPC code:

$$H=\begin{bmatrix}1&1&0&1&1&0&0&1&0&0\\0&1&1&0&1&1&1&0&0&0\\0&0&0&1&0&0&0&1&1&1\\1&1&0&0&0&1&1&0&1&0\\0&0&1&0&0&1&0&1&0&1\end{bmatrix}$$

First, we put H into *row-echelon form* (i.e. so that in any two successive rows that do not consist entirely of zeros, the leading 1 in the lower row occurs further to the right than the leading 1 in the higher row). The matrix H is put into this form by applying *elementary row operations* in GF (2), which are; interchanging two rows or adding one row to another modulo 2. From linear algebra we know that by using only elementary row operations the modified parity-check matrix will have the same codeword set as the original, (as the new system of linear equations will have an unchanged solution set). The 1-st and 2-nd columns of H already have ones on the diagonal and entries in these columns below the diagonal are removed by replacing the 4-throw with the modulo-2 sum of the 1-st and 4-th rows. The 3-rd column of H does not have a one on the diagonal but this can be obtained by swapping the 3-rd and 5-th rows. Finally, replacing the 5-th row with the modulo two sum of the 5-th and 4-th rows gives Hr in row-echelon form: Next the parity-check matrix is put into *reduced* row-echelon form (i.e. so that any column that contains a leading one has zeros everywhere else). The 1-st column is already correct and the entry in the 2-nd column above the diagonal is removed by replacing the 1-st row with the modulo-2 sum of the 1-st and 2-nd rows. Similarly the entry in the 3-nd column above the diagonal is removed by replacing the 2-nd row with the modulo-2 sum of the 2-nd and 3-rd rows. To clear the 4-th column the 1-st row is replace with the modulo-2 sum of the 1-st and 4-th rows. Finally, to clear the 5-th column involves adding the 5-th row to the 1-st, 2-nd and 4-th rows gives Hrr in reduced row-echelon form:

$$H_{rr} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 1 \end{bmatrix}.$$

Lastly, using column permutations we put the parity-check matrix into standard form (where the last m columns of Hstd are the m columns of Hrr which contain the leading ones)

$$H_{std} = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

In this final step column permutations have been used and so the codewords of Hstd will be permuted versions of the codewords corresponding to H. A solution is to keep track of the column permutation used to create Hstd, which in this case is

$$\Pi = \begin{bmatrix} 6 & 7 & 8 & 9 & 10 & 1 & 2 & 3 & 4 & 5 \end{bmatrix},$$

and apply the inverse permutation to each Hstd codeword before it is transmitted.

Alternatively, if the channel is memoryless, and so the order of codeword bits is unimportant, a far easier option is to apply to the original H to give a parity-check matrix

$$H' = \begin{bmatrix} 1 & 1 & 0 & 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

with the same properties as H but which shares the same codeword bit ordering as Hstd.

Finally, a generator G for the code with parity-check matrices Hstd and H′ is given by

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 1 \end{bmatrix}$$

All of this processing can be done off-line and just the matrices G and H′ provided to the encoder and decoder respectively. However, the drawback of this approach is that, unlike H, the matrix G will most likely not be sparse and so the matrix multiplication c = uG, at the encoder will have complexity in the order of n2 operations. As n is large for LDPC codes, from thousands to hundreds of thousands of bits, the encoder can become prohibitively complex. Later we will see that structured parity-check matrices can be used to significantly lower this implementation complexity, however for arbitrary parity-check matrices a good approach is to avoid constructing G at all and instead to encode using back substitution with H as is demonstrated in the following.

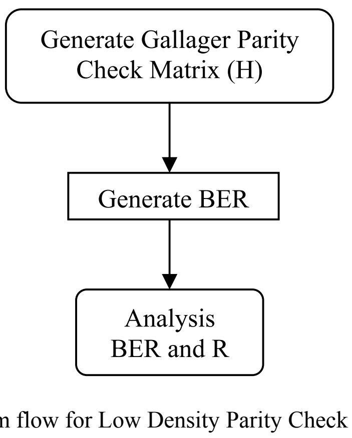
E. Decoding

There are two basic decoding schemes for the LDPC codes. In the first scheme, the decoder computes all parity checks and then changes any digit that is contained in more than some fixed number of unsatisfied parity- check equations. Using these new values, the parity checks are recomputed, and the process is repeated until the parity checks are all satisfied.

In the second scheme an algorithm known as the "sum-product" algorithm is used which is similar to the "belief-propagation algorithm" used in networks. Information about each bit of the codeword derived from the received data is expressed as a probability ratio, the probability of the bit being 1 divided by the probability of the bit being 0. The probability ratios will be adjusted to take account of information obtained from other bits along with the requirement that the parity checks be satisfied. The algorithm alternates between recalculating the probability ratios for each check.

III. Methodology

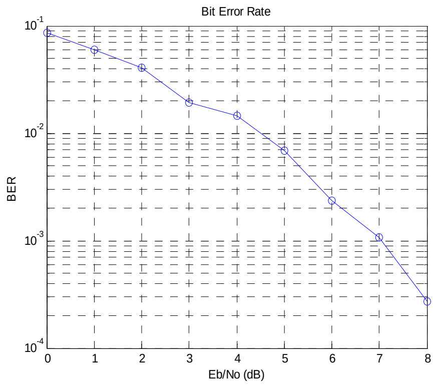
In this paper, Low Density Parity Check code was implemented using Math lab. Gallager (Regular) Parity Check Matrix approach is applied for this project. The theory of regular LDPC code has been explained in previous section. For this project, we have examined three codes with different parameter but having the same length. The first code with parameter N=20, K=10,wc=1 or matrix dimension 10x20.The second code with parameter N=20, K=5,wc=3.For the last code with parameter N=20, K=4,wc=4.The parity check matrix for the three codes have been created. After creating parity check of these three codes, the performances of those codes have been measured by determining value of BER and Code rates(R).



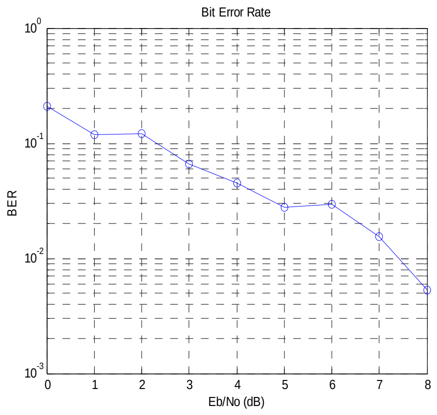
Algorithm flow for Low Density Parity Check and BER

IV. Result and Analysis

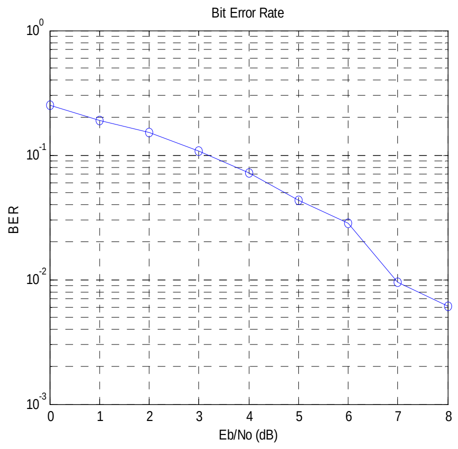
For N=20, K=10,wc=1 (10x20)



For N=20, K= 5, wc=3   15 x20)



For N=20, K= 4, wc= 4 (16 x20)



From the simulation result, we can compare the performances of those three codes based on the Bit Error Rate (BER), Code Rate(R) and SNR (Eb/No). For the first codes, the value of BER at 8dB is in the range between $10^{-3.5}$ and 10^{-4}.The second codes, the BER value at 8dB is in the range between $10^{-2.5}$ and 10^{-3}.Then the BER value for the third is in the range between 10^{-2} and $10^{-2.5}$.Meanwhile, the code rates of the first code is R= ½..The value of code rates for the second code is R= ¾ and the codes rate value for the third code is R=1/5. Generally the lower the code rate, the higher the coding gains. In other word, better Codes provides better coding gains and higher complexity. Therefore, the third code has better coding gain compared to others. But in tern of BER, the first code is the smallest value even though we have used the same length (N=20).

V. Conclusion

Low-density-parity-check codes have been studied a lot in the last years and huge progresses have been made in the understanding and ability to design iterative coding systems. The iterative decoding approach is already used in turbo codes but the structure of LDPC codes give even better results. In many cases they allow a higher code rate and also a lower error floor rate. In other achieve good coding gain performance, good LDPC code design is essential. A code design based on density evolution is only 0.0045dB away from the Shannon bound. However, it's a rate-1/2 irregular code with maximum variable degree of 100 and block size of 10^{7} bits. It also requires an average of more than 1000 iterations to achieve the result. The main disadvantage of these codes is that encoders are somehow more complex and that the code length has to be rather long to yield good results.

References

[1] R. Gallager, \Low-density parity-check codes," IRE Trans. Information Theory, pp. 21{28, January 1962.
[2] D. MacKay, \Good error correcting codes based on very sparse matrices," IEEE Trans. Information Theory, pp. 399{431, March 1999.
[3] J. L. Fan, Constrained coding and soft iterative decoding for storage. PhD thesis, Stanford University, 1999
[4] M. R. Soleymani, Yingzi Gao, U. Vilapornsawai, Turbo Coding for Satellite and Wireless Communications, Kluwer Academic Publishers, Massachusets, USA, 2002.

Othman O. Khalifa received his Bachelor's degree in Electronic Engineering from the Garyounis University, Libya in 1986. He obtained his Master degree in Electronics Science Engineering and PhD in Digital Image Processing from Newcastle University, UK in 1996 and 2000 respectively. He obtained his Master degree in Electronics Science Engineering and PhD in Digital Image Processing from Newcastle University, UK in 1996 and 2000 respectively. He worked in industrial for eight years and he is currently an Associate Professor and Head of the department of Electrical and Computer Engineering, International Islamic University Malaysia. His area of research interest is Communications, Information theory and Coding, Digital image / video and speech processing, coding and Compression, Wavelets, Fractal and Pattern Recognition. Dr Khalifa published more than 100 papers in international journals and Conferences. He is SIEEE member, IEEE computer, Image processing and Communication Society member.